22S:166

Computing in Statistics

Lecture 24
Nov. 2, 2016

Kate Cowles
374 SH
kate-cowles@uiowa.edu

Checking Values of Numeric Variables

- range checks
  - when you know what the range of possible values is for a given quantitative variable
- internal consistency checks
  - when you don't know in advance what a reasonable range is and simply want to look for extreme values relative to the rest of the data

Using proc means to inspect the data

- shows number of missing and nonmissing observations
- inspection of min and max values suggests there are errors and we need to do more checking

```
/****************************************************************************
Program 2-1  Using PROC MEANS to detect invalid and missing values
****************************************************************************
PROC MEANS DATA=PATIENTS N NMISS MIN MAX MAXDEC=3;
   TITLE "Checking Numeric Variables in PATIENTS data set";
   VAR HR SBP DBP;
RUN;
```

```
              Checking Numeric Variables in PATIENTS data set             1

                             The MEANS Procedure

                                           N
 Variable  Label                     N  Miss        Minimum        Maximum
 -----------------------------------------------------------------------
 HR        Heart Rate               28     3         10.000        900.000
 SBP       Systolic Blood Pressure  27     4         20.000        400.000
 DBP       Diastolic Blood Pressure 28     3          8.000        200.000
 -----------------------------------------------------------------------
```

Prettier data description using proc tabulate

- FORMAT option tells SAS to use numeric format 7.3 for all output in this procedure unless otherwise specified
  - field width of 7 with 3 places to the right of the decimal point
- VAR statement tells procedure which quantitative variables to produce summary statistics for
- content of TABLES statement
  - optional definition of separate pages of table goes before first comma if used (not used here)
  - definition of rows of table followed by comma
  - definition of columns of table
  - / optional additional options

- TABLES statement in this example
  - one row for each of the variables HR, SBP, and DBP
  - statistics N, NMISS, MEAN, MIN, and MAX in the columns
  - RTSPACE = 18 allows for 18 spaces for all row labels
  - N*F=7.0 NMISS*F=7.0 means to use format 7..0 for N and NMISS
- KEYLABEL statement replaces keywords for chosen statistics with more understandable lables
- `proc tabulate` output will not print correctly if you do not have the `formchar` option set

```
/************************************************************************
Program 2-2  Using PROC TABULATE to display descriptive data
*************************************************************************
PROC TABULATE DATA=PATIENTS FORMAT=7.3;
   TITLE "Statistics for Numeric Variables";
   VAR HR SBP DBP;
   TABLES HR SBP DBP,
          N*F=7.0 NMISS*F=7.0 MEAN MIN MAX / RTSPACE=18;
   KEYLABEL N     = 'Number'
            NMISS = 'Missing'
            MEAN  = 'Mean'
            MIN   = 'Lowest'
            MAX   = 'Highest';
RUN;

                     Statistics for Numeric Variables                    2

        ----------------------------------------------------------
        |                |Number |Missing| Mean  |Lowest |Highest|
        |----------------+-------+-------+-------+-------+-------|
        |Heart Rate      |     28|      3|107.393| 10.000|900.000|
        |----------------+-------+-------+-------+-------+-------|
        |Systolic Blood  |       |       |       |       |       |
        |Pressure        |     27|      4|144.519| 20.000|400.000|
        |----------------+-------+-------+-------+-------+-------|
        |Diastolic Blood |       |       |       |       |       |
        |Pressure        |     28|      3| 88.071|  8.000|200.000|
        ----------------------------------------------------------
```

Using PROC UNIVARIATE to look for outliers

- proc univariate yields more detailed and useful information about values of numeric variables
- PLOT option provides m
  - stem-and-leaf plot if dataset is fewer than 200 observations
    - ∗ very ugly histogram otherwise
  - box plot
  - normal probability plot

Using ODS statement and ID statement to pinpoint check for extreme values

- ID statement prints values of one variable, in addition to observation number, in table of extreme values
- ODS (output delivery system) statement can be used to limit which parts of PROC UNIVARIATE output are printed
  - available in Version 7 and later of SAS

```
/*****************************************************************************
Program 2-4  Adding an ID statement to PROC UNIVARIATE
*****************************************************************************
/************************************************************\
| The ODS statement is valid for V7 and above                |
| Note that the name EXTREMEOBS may change in future releases |
| Use ODS TRACE ON; before the PROC and ODS TRACE OFF; after  |
| the PROC to obtain a list of output object names (found in  |
| the SAS Log).                                               |
\************************************************************/

ODS SELECT EXTREMEOBS;

PROC UNIVARIATE DATA=PATIENTS PLOT;
   TITLE "Using PROC UNIVARIATE to look for Outliers";
   ID PATNO;
   VAR HR SBP DBP;
RUN;

                Using PROC UNIVARIATE to look for Outliers               11
                         The UNIVARIATE Procedure
                Variable:  DBP  (Diastolic Blood Pressure)

                           Extreme Observations

           --------Lowest--------        --------Highest-------

           Value   PATNO       Obs        Value   PATNO       Obs

               8   020          23          106   027          28
              20   011          12          120   004           4
              64   013          14          120   010          11
              68   025          27          180   009          10
              68   006           6          200   321          22
```

Using PROC PRINT with a WHERE statement to list invalid data values

- WHERE statement may have multiple conditions
- note logical operators used

```
/*****************************************************************************
Program 2-5  Using a WHERE statement with PROC PRINT to list out-of-range da
*****************************************************************************
PROC PRINT DATA=PATIENTS;
   WHERE (HR NOT BETWEEN 40 AND 100 AND HR IS NOT MISSING)      OR
         (SBP NOT BETWEEN 80 AND 200 AND SBP IS NOT MISSING)    OR
         (DBP NOT BETWEEN 60 AND 120 AND DBP IS NOT MISSING);
   TITLE "Out-of-range Values for Numeric Variables";
   ID PATNO;
   VAR HR SBP DBP;
RUN;

                 Out-of-range Values for Numeric Variables              12

                        PATNO     HR    SBP    DBP

                         004     101    200    120
                         008     210      .      .
                         009      86    240    180
                         010       .     40    120
                         011      68    300     20
                         014      22    130     90
                         017     208      .     84
```

```
                         321     900    400    200
                         020      10     20      8
                         023      22     34     78
```

Checking for Missing Values

- ways in which missing values can occur in a SAS data set
  - raw data value missing (intentionally or accidentally)
  - invalid value can cause missing value to be created
    - e.g. reading charcter value with a numeric informat
    - invalid dates
  - operations, such as assignment statements, can create missing values
    - e.g. trying to take log of negative number
- for some variables, missing values may be expected and may not create problems in analysis
- for other variables (such as patient IDs), missing values may not be permissible

Inspecting the SAS log

- if you know that a numeric variable in the data file contains invalid character values, might want to read that variable with a character informat and perform character to numeric conversion using INPUT function
  - keeps log file more readable
  - makes it easier to spot other errors reported in log

```
8              OPTIONS FORMCHAR = "|----|+|---+=|-/\<>*"
9                  LS = 75   NODATE;
10
11         * LIBNAME CLEAN "C:\CLEANING";
12
13         *DATA CLEAN.PATIENTS;
14         DATA PATIENTS;
15            *INFILE "C:\temp\patients.dat" PAD;
16            INFILE "/group/ftp/pub/kcowles/datasets/patients.dat" PAD;
17            INPUT @1  PATNO    $3.
18                  @4  GENDER   $1.
19                  @5  VISIT    MMDDYY10.
20                  @15 HR       3.
21                  @18 SBP      3.
22                  @21 DBP      3.
23                  @24 DX       $3.
24                  @27 AE       $1.;
25
26            LABEL PATNO   = "Patient Number"
27                  GENDER  = "Gender"
28                  VISIT   = "Visit Date"
29                  HR      = "Heart Rate"
30                  SBP     = "Systolic Blood Pressure"
31                  DBP     = "Diastolic Blood Pressure"
32                  DX      = "Diagnosis Code"
33                  AE      = "Adverse Event?";
34
35            FORMAT VISIT MMDDYY10.;
36
37         RUN;

NOTE: The infile "/group/ftp/pub/kcowles/datasets/patients.dat" is:
      File Name=/group/ftp/pub/kcowles/datasets/patients.dat,
      Owner Name=UNKNOWN,Group Name=UNKNOWN,
```

```
      Access Permission=rw-r--r--,
      File Size (bytes)=867

NOTE: Invalid data for VISIT in line 7 5-14.
RULE:     ----+----1----+----2----+----3----+----4----+----5----+----6----+
7         007M08/32/1998 88148102   0
      66
     131
     196
PATNO=007 GENDER=M VISIT=. HR=88 SBP=148 DBP=102 DX=  AE=0 _ERROR_=1 _N_=7
NOTE: Invalid data for VISIT in line 12 5-14.
12        011M13/13/1998 68300 20  41
      66
     131
     196
PATNO=011 GENDER=M VISIT=. HR=68 SBP=300 DBP=20 DX=4 AE=1 _ERROR_=1 _N_=12
NOTE: Invalid data for VISIT in line 21 5-14.
21        123M15/12/1999 60        10
      66
     131
     196
PATNO=123 GENDER=M VISIT=. HR=60 SBP=. DBP=. DX=1 AE=0 _ERROR_=1 _N_=21
NOTE: Invalid data for VISIT in line 23 5-14.
23        020F99/99/9999 10 20  8   0
      66
     131
     196
PATNO=020 GENDER=F VISIT=. HR=10 SBP=20 DBP=8 DX=  AE=0 _ERROR_=1 _N_=23
NOTE: Invalid data for VISIT in line 28 5-14.
NOTE: Invalid data for HR in line 28 15-17.
28        027FNOTAVAIL  NA 166106  70
      66
     131
     196
```

```
PATNO=027 GENDER=F VISIT=. HR=. SBP=166 DBP=106 DX=7 AE=0 _ERROR_=1 _N_=28
NOTE: 31 records were read from the infile
      "/group/ftp/pub/kcowles/datasets/patients.dat".
      The minimum record length was 26.
      The maximum record length was 27.
NOTE: The data set WORK.PATIENTS has 31 observations and 8 variables.
NOTE: DATA statement used:
      real time           0.18 seconds
      cpu time            0.08 seconds
```

Before using dataset for any serious purpose,

- invalid dates need to be checked
- decision needs to be made concerning 'NA' value for heart rate

Using proc means and proc freq to count missing values

- using proc means to check for missing numeric values is straightforward

```
TITLE "Missing Value Check for the PATIENTS data set";

PROC MEANS DATA=PATIENTS N NMISS;
RUN;
```

```
               Missing Value Check for the PATIENTS data set              1

                            The MEANS Procedure

                                                               N
            Variable    Label                         N     Miss
            ------------------------------------------------------
            VISIT       Visit Date                   24        7
            HR          Heart Rate                   28        3
            SBP         Systolic Blood Pressure      27        4
            DBP         Diastolic Blood Pressure     28        3
            ------------------------------------------------------
```

Using proc freq to count missing values of character variables

- may not be sensible to create one-way frequency tables for all character variables
  - some may take on thousands of unique values
- create character format that has only two values, one for missing and one for nonmissing
- to make proc freq display output for all character values in dataset, use SAS keyword `_CHARACTER_` in the tables statement (or provide list of names of character variables)

```
PROC FORMAT;
   VALUE $MISSCNT ' '   = 'MISSING'
                  OTHER = 'NONMISSING';
RUN;

PROC FREQ DATA=PATIENTS;
   TABLES _CHARACTER_ / NOCUM MISSING;
   FORMAT _CHARACTER_ $MISSCNT.;
RUN;
```

```
               Missing Value Check for the PATIENTS data set              2

                            The FREQ Procedure

                              Patient Number

                    PATNO         Frequency     Percent
                    -----------------------------------
                    MISSING              1        3.23
                    NONMISSING          30       96.77


                                  Gender

                    GENDER        Frequency     Percent
                    -----------------------------------
                    MISSING              1        3.23
                    NONMISSING          30       96.77


                              Diagnosis Code

                    DX            Frequency     Percent
                    -----------------------------------
                    MISSING              8        25.81
                    NONMISSING          23        74.19


                              Adverse Event?

                    AE             Frequency     Percent
                    -----------------------------------
                    MISSING              1         3.23
                    NONMISSING          30        96.77
```

o

List Missing data values and ID variable

- counting missing values usually is not enough
- if you have variables for which missing vlaues are not allowed, you need to locate the observations so that the original data values can be checked and corrected
- using the SAS internal name `_NULL_` in a `DATA` statement causes SAS not to actually create another dataset in its memory
  - used when we want SAS to send output to a file or display
- `PUT` statement sends row of output to file or display

```
/****************************************************************************
Program 3-2  Writing a simple Data Step to list missing data values and an I
Variable
****************************************************************************
DATA _NULL_;
   INFILE "/group/ftp/pub/kcowles/datasets/patients.dat" PAD;
   *INFILE "C:\CLEANING\PATIENTS.TXT" PAD;
   FILE PRINT; ***Send output to the output window;
   TITLE "Listing of Missing Values";
   ***Note: We will only input those variables of interest;
    INPUT @1  PATNO    $3.
        @5  VISIT    MMDDYY10.
        @15 HR       3.
        @27 AE       $1.;
   IF VISIT = .  THEN PUT "Missing or invalid visit date for ID " PATNO;
   IF HR    = .  THEN PUT "Missing or invalid HR for ID " PATNO;
   IF AE    = ' ' THEN PUT "Missing or invalid AE for ID " PATNO;
RUN;

                         Listing of Missing Values                         3
Missing or invalid visit date for ID 007
Missing or invalid HR for ID 010
Missing or invalid visit date for ID 011
Missing or invalid AE for ID 013
Missing or invalid visit date for ID 015
Missing or invalid visit date for ID 123
Missing or invalid visit date for ID 321
Missing or invalid visit date for ID 020
Missing or invalid visit date for ID 027
Missing or invalid HR for ID 027
Missing or invalid HR for ID 029
```

Attempting to locate missing or invalid unique identifiers

- obviously you can't list which patient number is missing!
- one solution: report the patient number or numbers preceding the missing number *(in the original order of the raw data file*
- if you sort the dataset first, all missing values will "float" to the top and you won't have a clue which patients they belong ti
- add the observation number to the output by printing the value of the internal SAS variable `_N_`

```
/***************************************************************************
Program 3-3  Attempting to locate a missing or invalid patient ID by listing
two previous ID's
***************************************************************************
DATA _NULL_;
   SET PATIENTS;
   ***Be sure to run this on the unsorted data set;
   FILE PRINT;
   TITLE "Listing of Missing Patient Numbers";
   PREV_ID = LAG(PATNO);
   PREV2_ID = LAG2(PATNO);
   IF PATNO = ' ' THEN PUT "Missing Patient ID. Two previous ID's are:"
      PREV2_ID "and " PREV_ID / @5 "Missing Record is number " _N_;
   ELSE IF INPUT(PATNO,?? 3.)  = . THEN
      PUT "Invalid Patient ID:" PATNO +(-1)". Two previous ID's are:"
      PREV2_ID "and " PREV_ID / @5 "Missing Record is number " _N_;
RUN;
```

```
                    Listing of Missing Patient Numbers                    4
Invalid Patient ID:XX5. Two previous ID's are:003 and 004
    Missing Record is number 5
Missing Patient ID. Two previous ID's are:006 and 007
    Missing Record is number 8
```

The +(-1) pointer control moves the pointer backward to remove the unwanted blank that occurs between the value of PATNO and the period.